

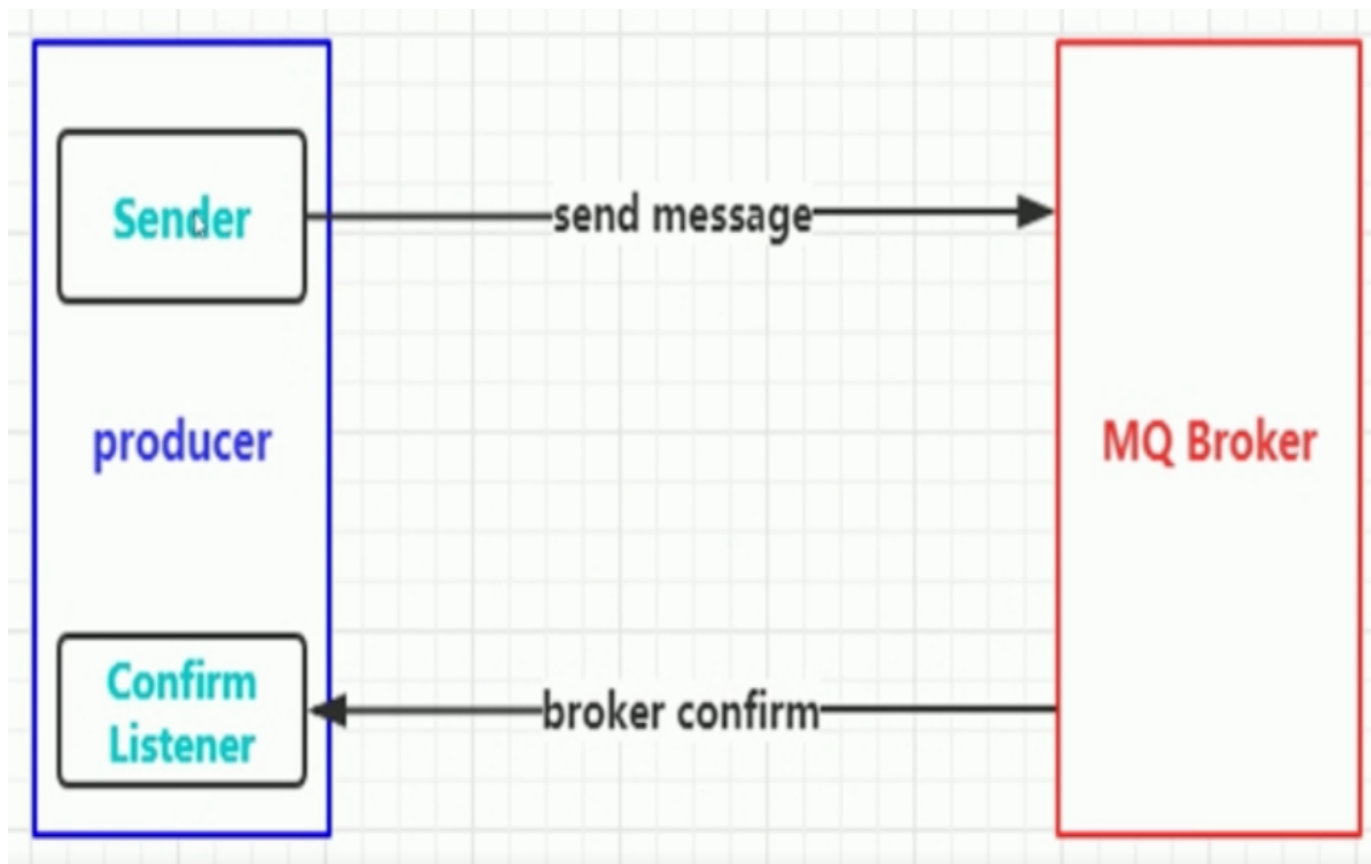
3.3) 消息的确认Confirm

一:理解Confirm消息确认机制

1.1)消息的确认是指，生产端投递消息后，若mq-server接受到消息，就会给生产者一个应答

1.2) 生产端根据mq broker返回应答来确认该条消息是否正常发送到了broker,这种方式是消息可靠性投递的核心保障

1.3) 消息确认机制的流程图



1.4) 如何来做消息的confirm

第一步:在channel上开启确认模式 `channel.confirmSelect();`

第二部:在channel上增加confirm监听，来监听成功和异常的confirm结果

1.5)代码演示 生产者:

```
public static void main(String[] args) throws IOException, TimeoutException {
    ConnectionFactory connectionFactory = new ConnectionFactory();
    connectionFactory.setVirtualHost("cloudmall");
    connectionFactory.setHost("47.104.128.12");
    connectionFactory.setPort(5672);
```

```

Connection connection = connectionFactory.newConnection();
Channel channel = connection.createChannel();
//开启confirm
channel.confirmSelect();
channel.addConfirmListener(new ConfirmListener() {
    /**
     * 接口成功
     *
     * @param deliveryTag deliveryTag 消息id
     * @param multiple 是否批量
     * @throws IOException
     */
    @Override
    public void handleAck(long deliveryTag, boolean multiple) throws IOException {
        System.out.println("消息id" + deliveryTag + ".....ack");
    }
    @Override
    public void handleNack(long deliveryTag, boolean multiple) throws IOException {
        System.out.println("消息id" + deliveryTag + ".....no ack");
    }
});
channel.basicPublish("test.confirm.exchange", "test.confirm.key", null, "confirm消息".getBytes());
}

```

消费者:

```

public static void main(String[] args) throws IOException, TimeoutException, InterruptedException {
    ConnectionFactory connectionFactory = new ConnectionFactory();
    connectionFactory.setVirtualHost("cloudmall");
    connectionFactory.setHost("47.104.128.12");
    connectionFactory.setPort(5672);

    Connection connection = connectionFactory.newConnection();
    Channel channel = connection.createChannel();
    channel.exchangeDeclare("test.confirm.exchange", "topic", true, false, null);
    channel.queueDeclare("test.confirm.queue", true, false, false, null);
    channel.queueBind("test.confirm.queue", "test.confirm.exchange", "test.confirm.#");
    QueueingConsumer queueingConsumer = new QueueingConsumer(channel);

    channel.basicConsume("test.confirm.queue", true, queueingConsumer);
    while (true) {
        QueueingConsumer.Delivery delivery = queueingConsumer.nextDelivery();
        System.out.println(new String(delivery.getBody()));
    }
}

```

